A Comparative Study of Energy Minimization Methods for Markov Random Fields

Richard Szeliski[1], Ramin Zabih[2], Daniel Scharstein[3], Olga Veksler[4], Vladimir Kolmogorov[5], Aseem Agarwala[6], Marshall Tappen[7], and Carsten Rother[1]

[1] Microsoft Research
{szeliski, carrot}@microsoft.com
[2] Cornell University
rdz@cs.cornell.edu
[3] Middlebury College
schar@middlebury.edu
[4] University of Western Ontario
olga@csd.uwo.ca
[5] University College London
vnk@adastral.ucl.ac.uk
[6] University of Washington
aseem@cs.washington.edu
[7] MIT
mtappen@mit.edu

Abstract. One of the most exciting advances in early vision has been the development of efficient energy minimization algorithms. Many early vision tasks require labeling each pixel with some quantity such as depth or texture. While many such problems can be elegantly expressed in the language of Markov Random Fields (MRF's), the resulting energy minimization problems were widely viewed as intractable. Recently, algorithms such as graph cuts and loopy belief propagation (LBP) have proven to be very powerful: for example, such methods form the basis for almost all the top-performing stereo methods. Unfortunately, most papers define their own energy function, which is minimized with a specific algorithm of their choice. As a result, the tradeoffs among different energy minimization algorithms are not well understood. In this paper we describe a set of energy minimization benchmarks, which we use to compare the solution quality and running time of several common energy minimization algorithms. We investigate three promising recent methods—graph cuts, LBP, and tree-reweighted message passing—as well as the well-known older iterated conditional modes (ICM) algorithm. Our benchmark problems are drawn from published energy functions used for stereo, image stitching and interactive segmentation. We also provide a general-purpose software interface that allows vision researchers to easily switch between optimization methods with minimal overhead. We expect that the availability of our benchmarks and interface will make it significantly easier for vision researchers to adopt the best method for their specific problems. Benchmarks, code, results and images are available at http://vision.middlebury.edu/MRF.

A. Leonardis, H. Bischof, and A. Prinz (Eds.): ECCV 2006, Part II, LNCS 3952, pp. 16–29, 2006.

1 Introduction

Many problems in early vision involve assigning each pixel a label, where the labels represent some local quantity such as disparity. Such pixel labeling problems are naturally represented in terms of energy minimization, where the energy function has two terms: one term penalizes solutions that are inconsistent with the observed data, while the other term enforces spatial coherence. One of the reasons this framework is so popular is that it can be justified in terms of maximum a posteriori estimation of a Markov Random Field [1, 2]. Despite the elegance and power of the energy minimization approach, its early adoption was slowed by computational considerations. The algorithms that were originally used, such as ICM [1] or simulated annealing [3], proved to be extremely inefficient.

In the last few years, energy minimization approaches have had a renaissance, primarily due to powerful new optimization algorithms such as graph cuts [4, 5] and loopy belief propagation (LBP) [6]. The results, especially in stereo, have been dramatic; according to the widely-used Middlebury stereo benchmarks [7], almost all the top-performing stereo methods rely on graph cuts or LBP. Moreover, these methods give substantially more accurate results than were previously possible. Simultaneously, the range of applications of pixel labeling problems has also expanded dramatically, moving from early applications such as image restoration [1], texture modeling [8], image labeling [9], and stereo matching [3, 4], to applications such as interactive photo segmentation [10, 11] and the automatic placement of seams in digital photomontages [12].

Relatively little attention has been paid, however, to the relative performance of various optimization algorithms. Among the few exceptions are [14], which compared the efficiency of several different max flow algorithms for graph cuts, and [13], which compared graph cuts with LBP. [13] also noted a particulary impressive demonstration of the effectiveness of modern energy minimization methods: for the stereo problems in the Middlebury benchmarks, both graph cuts and LBP produced results whose energy is lower than the ground truth solution. We will return to this issue at the end of this paper.

While it is generally accepted that algorithms such as graph cuts are a huge improvement over older techniques such as simulated annealing, less is known about the efficiency vs. accuracy tradeoff among more recently developed algorithms. Concurrently with our work, [15] compared tree-reweighted message passing, LBP and graph cuts for highly connected graphs.

In this paper, we evaluate a number of different energy minimization algorithms for pixel labeling problems. We propose a number of benchmark problems for energy minimization and use these benchmarks to compare several different energy minimization methods. Since much of the work in energy minimization has been motivated by pixel labeling problems over 2D grids, we have restricted our attention to problems with this simple topology. (The extension of our work to more general topologies, such as 3D, is straightforward.)

This paper is organized as follows. In section 2 we give a precise description of the energy functions that we consider, and present a simple but general software interface to describe such energy functions and to call an arbitrary energy

minimization algorithm. In section 3 we describe the different energy minimization algorithms that we have implemented, and in section 4 we present our set of benchmarks. In section 5 we provide our experimental comparison of the different energy minimization methods, and we conclude in section 6.

2 Problem Formulation and Experimental Infrastructure

We define a pixel labeling problem as assigning to every pixel p a label, which we write as l_p. The collection of all pixel-label assignments is denoted by l, the number of pixels is n, and the number of labels is m. The energy function E, which can also be viewed as the log likelihood of the posterior distribution of a Markov Random Field [2, 16], is composed of a data energy E_d and smoothness energy E_s, $E = E_d + \lambda E_s$. The data energy is simply the sum of a set of per-pixel data costs $d_p(l)$, $E_d = \sum_p d_p(l_p)$. In the MRF framework, the data energy comes from the (negative) log likelihood of the measurement noise.

We assume that pixels form a 2D grid, so that each p can also be written in terms of its coordinates $p = (i, j)$. We use the standard 4-connected neighborhood system, so that the smoothness energy is the sum of spatially varying horizontal and vertical nearest-neighbor smoothness costs, $V_{pq}(l_p, l_q)$, where if $p = (i, j)$ and $q = (s, t)$ then $|i - s| + |j - t| = 1$. If we let $\mathcal{N}$ denote the set of all such neighboring pixel pairs, the smoothness energy is

$$E_s = \sum_{\{p,q\} \in \mathcal{N}} V_{pq}(l_p, l_q). \tag{1}$$

Note that in equation 1, the notation $\{p, q\}$ stands for an unordered set, that is the sum is over *unordered* pairs of neighboring pixels.

In the MRF framework, the smoothness energy comes from the negative log likelihood of the prior. In this paper, we consider a general form of the smoothness costs, where different pairings of adjacent labels can lead to different costs. This is important in a number of applications, ranging from stereo matching (§8.2 of [4]) to image stitching and texture quilting [12, 17, 18].

A more restricted form of the smoothness energy is $E_s = \sum_{\{p,q\} \in \mathcal{N}} w_{pq} \cdot V(|l_p - l_q|)$, where the smoothness terms are the product of spatially varying per-pairing weights w_{pq} and a non-decreasing function of the label difference $V(\Delta l) = V(|l_p - l_q|)$. While we could represent V using an m-valued look-up table, for simplicity, we instead parameterize V using a simple clipped monomial form $V(\Delta l) = \min(|\Delta l|^k, V_{\max})$, with $k \in \{1, 2\}$. If we set $V_{\max} = 1.0$, we get the Potts model, $V(\Delta l) = 1 - \delta(\Delta l)$, which penalizes any pair of different labels uniformly (δ is the unit impulse function).

While they are not our primary focus, a number of important special cases have fast exact algorithms. If there are only two labels, the natural Potts model smoothness cost can be solved exactly with graph cuts (this was first applied to images by [19]). If the labels are the integers starting with 0 and the smoothness cost is an arbitrary convex function, [20] gives a graph cut construction. An algorithm due to [21] can be used with $V(\Delta l) = \Delta l$ (L_1 smoothness) and convex

data costs. However, the NP-hardness result proved in [4] applies if there are more than two labels, as long as the class of smoothness costs includes the Potts model. This, unfortunately, implies that the vast majority of MRF-based energy functions are intractable.

The class of energy functions we are considering is quite broad, and not all energy minimization methods can handle the entire class. For example, acceleration techniques based on distance transforms [22] can significantly speed up message-passing algorithms such as LBP or TRW, yet these methods are only applicable for certain smoothness costs V. Other algorithms, such as graph cuts, only have good theoretical guarantees for certain choices of V (see section 3 for a discussion of this issue). We will assume that any algorithm can run on any benchmark problem; this can generally be ensured by reverting to a weaker or slower of the algorithm if necessary for a particular benchmark.

2.1 Software Interface for Energy Minimization

Now that we have defined the class of energy functions that we minimize, we need to compare different energy minimization methods on the same energy function E. Conceptually, it is easy to switch from one energy minimization method to another, but in practice, most applications are tied to a particular choice of E. As a result, almost no one in vision has ever answered questions like "how would your results look if you used LBP instead of graph cuts to minimize your E?" (The closest to this was [13], who compared LBP and graph cuts for stereo.) In order to create a set of benchmarks, it was necessary to design a standard software interface (API) that allows a user to specify an energy function E and to easily call a variety of energy minimization methods to minimize E.

The software API is available at `http://vision.middlebury.edu/MRF`, as are all of our benchmarks and implementations of most of the energy minimization methods discussed in this paper. The API allows the user to define any energy function described above. The data cost energy can be specified implicitly, as a function $d_p()$ or explicitly as an array. The smoothness cost likewise can be specified either by defining the parameters k and $V_{\max}$, or by providing an explicit function or array. Excerpts from an example program that uses our API to call two different energy minimization algorithms on the same energy function are given below.

```
// Abstract definition of an energy function E
EnergyFunction *E = (EnergyFunction *) new EnergyFunction(data,smooth);
// Energy minimization of E via ICM
solver = (MRF *) new ICM(width,height,num_labels,E);
// To use graph cuts to minimize E instead, substitute the line below
// solver = (MRF *) new Expansion(width,height,num_labels,E);
// Run one iteration, store the amount of time it takes in t
solver->optimize(1,&t);
// Print out the resulting energy and running time
print_stats( solver->totalEnergy(), t);
```

Note that the interface also has the notion of an iteration, but it is up to each energy minimization method to interpret this notion. Most algorithms have some natural intermediate point where they have a current answer. By supporting this, our API allows us to plot the curve of energy versus time. This is particularly important because a number of powerful methods (such as TRW and graph cuts) make very small changes in the last few iterations.

2.2 Evaluation Methodology

To evaluate the quality of a given solution, we need the final energy E along with the computation time required, as a function of the number of iterations. For every benchmark, we produce a plot that keeps track of the energy vs. computation time for every algorithm tested. We implemented the algorithms in C or C++, and ran the benchmarks on a modern Pentium 4. Most of the experiments used the same machine (3.4 GHz, 2GB RAM), while a few used a fairly similar computer.

Of course, not all authors spent the same amount of effort tuning their implementation for our benchmarks. Note that while the natural way to compare energy minimization algorithms is in terms of their energy and speed, it is not always the case that the lowest energy solution is the best one for a vision problem. (We return to this issue at the end of section 6.)

3 Energy Minimization Algorithms

In this section, we describe the optimization algorithms that we have implemented and included in our interface. Most of the energy minimization algorithms were implemented by their original inventors; the exceptions are ICM and LBP, which we implemented ourselves (for LBP, we received help from several experts).

Iterated conditional modes (ICM) — Iterated conditional modes [1] uses a deterministic "greedy" strategy to find a local minimum. It starts with an estimate of the labeling, and then for each pixel it chooses the label giving the largest decrease of the energy function. This process is repeated until convergence, which is guaranteed to occur, and in practice is very rapid.

Unfortunately, the results are extremely sensitive to the initial estimate, especially in high-dimensional spaces with non-convex energies (such as arise in vision) due to the huge number of local minima. In our experiments, we initialized ICM in a winner-take-all manner, by assigning each pixel the label with the lowest data cost. This resulted in significantly better performance.

Graph cuts — The two most popular graph cuts algorithms, called the *swap move* algorithm and the *expansion move* algorithm, were introduced in [4]. These algorithms rapidly compute a strong local minimum, in the sense that no "permitted move" will produce a labeling with lower energy.

For a pair of labels α, β, a swap move takes some subset of the pixels currently given the label α and assigns them the label β, and vice-versa. The swap move

algorithm finds a local minimum such that there is no swap move, for any pair of labels α,β, that will produce a lower energy labeling. Analogously, we define an expansion move for a label α to increase the set of pixels that are given this label. The expansion move algorithm finds a local minimum such that no expansion move, for any label α, yields a labeling with lower energy.

The criteria for a local minimum with respect to expansion moves (swap moves) are so strong that there are many fewer minima in high dimensional spaces compared to standard moves. In the original work of [4] the swap move algorithm was shown to be applicable to any energy where V_{pq} is a semi-metric, and the expansion move algorithm to any energy where V_{pq} is a metric. The results of [5] imply that the expansion move algorithm can be used if for all labels α,β,and γ, $V_{pq}(\alpha,\alpha) + V_{pq}(\beta,\gamma) \leq V_{pq}(\alpha,\gamma) + V_{pq}(\beta,\alpha)$. The swap move algorithm can be used if for all labels α,β $V_{pq}(\alpha,\alpha) + V_{pq}(\beta,\beta) \leq V_{pq}(\alpha,\beta) + V_{pq}(\beta,\alpha)$. (This constraint comes from the notion of regular, i.e. submodular, binary energy functions, which are closely related to graph cuts.)

If the energy does not obey these constraints, graph cut algorithms can still be applied by "truncating" the violating terms [24]. In this case, however, we are no longer guaranteed to find the optimal labeling with respect to swap (or expansion) moves. In paractice, the performance of this version seems to work well when only relatively few terms need to be truncated.

Max-product loopy belief propagation (LBP) — To evaluate the performance of LBP, we implemented the max-product LBP version, which is designed to find the lowest energy solution. The other main variant of LBP, the sum-product algorithm, does not directly search for a minimum energy solution, but instead computes the marginal probability distribution of each node in the graph. The belief propagation algorithm was originally designed for graphs without cycles [25], in which case it produces the exact result for our energy. However, there is nothing in the formulation of BP that prevents it from being tried on graphs with loops.

In general, LPB is not guaranteed to converge, and may go into an infinite loop switching between two labelings. Felzenszwalb and Huttenlocher [22] present a number of ways to speed up the basic algorithm. In particular, our LBP implementation uses the distance transform method described in [22], which significantly reduces the running time of the algorithm.

Tree-reweighted message passing (TRW) — Tree-reweighted message passing [30] is a message-passing algorithm similar, on the surface, to LBP. Let $M^t_{p\to q}$ be the message that pixel p sends to its neighbor q at iteration t; this is a vector of size m (the number of labels). The message update rule is:

$$M^t_{p\to q}(l_q) = \min_{l_p} \left(c_{pq}\{d_p(l_p) + \sum_{s\in\mathcal{N}(p)} M^{t-1}_{s\to p}(l_p)\} - M^{t-1}_{q\to p}(l_p) + V_{pq}(l_p,l_q) \right).$$

The coefficients c_{pq} are determined in the following way. First, a set of trees from the neighborhood graph (a 2D grid in our case) is chosen so that each edge is in at least one tree. A probability distribution ρ over the set of trees is

then chosen. Finally, c_{pq} is set to ρ_{pq}/ρ_p, i.e. the probability that a tree chosen randomly under ρ contains edge (p, q) *given* that it contains p. Note that if c_{pq} were set to 1, the update rule would be identical to that of standard LBP.

An interesting feature of the TRW algorithm is that it computes a lower bound on the energy. The original TRW algorithm does not necessarily converge, and does not, in fact, guarantee that the lower bound always increases with time. In this paper we use an improved version of TRW due to [23], which is called sequential TRW, or TRW-S. In this version, the lower bound estimate is guaranteed not to decrease, which results in certain convergence properties. In TRW-S we first select an arbitrary pixel ordering function $S(p)$. The messages are updated in order of increasing $S(p)$ and at the next iteration in the reverse order. Trees are constrained to be chains that are monotonic with respect to $S(p)$. Note that the algorithm can be implemented using half as much memory as standard BP [23].

4 Benchmark Problems

For our benchmark problems, we have created a representative set of low-level energy minimization problems drawn from a range of different applications. As with the optimization methods, we were fortunate enough to persuade the original authors of the problems to contribute their energy functions and data.

Stereo matching — For stereo matching, we followed in the footsteps of [13] and used a simple energy function for stereo, applied to images from the widely-used Middlebury stereo data set [7]. We used different energy functions for different images, to make the optimization problems more varied. For the "Tsukuba" image we used the truncated L_1 distance $V_{\max} = 2, k = 1$, with $\lambda = 20$ and $m = 16$ labels. For "Venus" we used the truncated L_2 distance $V_{\max} = 2, k = 7$, with $\lambda = 50$ and $m = 20$ labels. For "Teddy" we used the Potts model $V_{\max} = 1, k = 1$, with $\lambda = 10$ and $m = 60$ labels. The default smoothness weight was $w_{pq} = 1$ at all pixels. For "Tsukuba" and "Teddy" we increased the weight at locations where the intensity gradient g_{pq} in the left image is small: we used $w_{pq} = 2$ if $|g_{pq}| \leq 8$ for "Tsukuba," and $w_{pq} = 3$ if $|g_{pq}| \leq 10$ for "Teddy."

Photomontage — The Photomontage system [12] seamlessly stitches together multiple photographs for a variety of photo merging applications. We formed benchmarks for two such applications, panoramic stitching and group photo merging. The input is a set of aligned images $S_1, S_2, \ldots, S_m$ of equal dimension; the labels are the image indexes, i.e. $1, 2, ..., m$; the final output image is formed by copying colors from the input images according to the computed labeling. If two neighbors p and q are assigned the same input image, they should appear natural in the composite and so $V_{pq}(i, i) = 0$. If $l_p \neq l_q$, we say that a seam exists between p and q; then V_{pq} measures how visually noticeable the seam is in the composite. The data term $d_p(i)$ is 0 if pixel p is in the field of view of image i, and ∞ otherwise.

The first benchmark stitches together the panorama in Fig. 8 of [12]. (See the project web page for all images.) The smoothness energy, derived from [18], is

$V_{pq} = |S_{l_p}(p) - S_{l_q}(p)| + |S_{l_p}(q) - S_{l_q}(q)|$. This energy function is suitable for the expansion algorithm without truncation.

The second benchmark stitches together the five group photographs shown in Fig. 1 of [12]. The best depiction of each person is to be included in a composite. Photomontage itself is interactive, but to make the benchmark repeatable the user strokes are saved into a data file. For any pixel p underneath a drawn stroke, $d_p(l_p) = 0$ if l_p equals the user-indicated source image, and ∞ otherwise. The smoothness terms are modified from the first benchmark to encourage seams along strong edges. The expansion algorithm is applicable to this energy only after truncating certain terms, but it continues to work well in practice.

Binary image segmentation — Binary MRF's are also widely used in medical image segmentation [10], stereo matching using minimal surfaces [27, 28], and video segmentation using stereo disparity cues [29] As previously mentioned, for the natural Ising model smoothness costs, the global minimum can be computed rapidly via a graph cuts [19]; this result has been generalized to other smoothness costs by [5].) Nevertheless, such energy functions still form an interesting benchmark, since there may well be other heuristic algorithms that perform faster while achieving nearly the same level of performance.

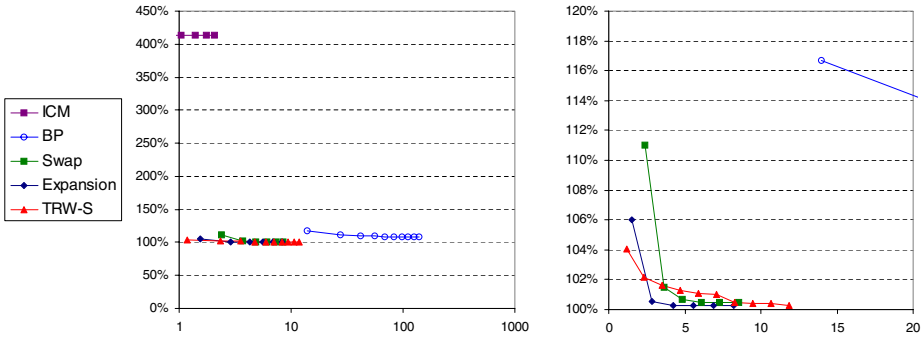
Our benchmark consists of a segmentation problem, inspired by the interactive segmentation algorithm of [10] or its more recent extensions [11]. As with our Photomontage stitching example, this application requires user interaction; we handle this issue as above, by saving the user interactions to a file and using them to derive the data costs.

The data cost is the log likelihood of a pixel belonging to either foreground or background and is modeled as two separate Gaussian mixture models as in [11]. The smoothness term is a standard Potts model which is contrast sensitive: $V_{pq} = ||\exp(-\beta\|x_i - x_j\|^2)|| + \lambda_2$, where $\lambda = 50$ and $\lambda_2 = 10$. The quantity β is set to $(2\langle\|x_i - x_j\|^2\rangle)^{-1}$ where the expectation denotes an average over the image, as motivated in [11]. The impact of λ_2 is to remove small and isolated areas that have high contrast.

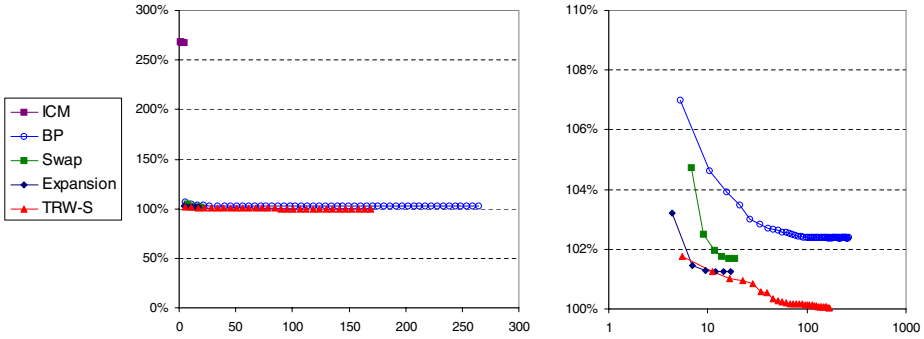
Image restoration and inpainting — We experimented with the "penguin" image, which appears in figure 7 in [22]. We added random noise to each pixel, and also obscured a portion of the image. The labels are intensities, and the data cost for each pixel is the squared difference between the label and the observed distance. However, pixels in the obscured portion have a data cost of 0 for any intensity. The smoothness energy was the truncated L_2 distance with uniform w_{pq}'s (we used $V_{\max} = 200, k = 2, w_{pq} = 25$).

5 Experimental Results

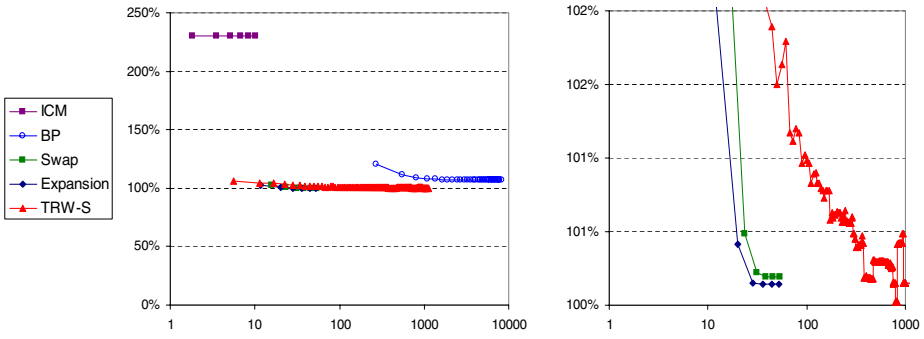
The experimental results from running the different optimization algorithms on these benchmarks are given in figure 1 (stereo), figure 2 (Photomontage), and figure 3 (binary image segmentation). The images themselves are provided on the project web page. The x-axis of these plots shows running times, measured



"Tsukuba" energy, with the truncated L_1 distance for V



"Venus" energy, with the truncated L_2 distance for V



"Teddy" energy, with the Potts model distance for V

Fig. 1. Results on stereo matching benchmarks. Each plot shows energies vs. run time in seconds. Energies are given relative to the largest lower bound provided by the TRW-S method. The plots on the right are zoomed versions of the plots on the left. Note that some of the horizontal (time) axes use a log scale to better visualize the results. ICM is omitted in the right plots, due to its poor performance. Depth map images are available at `http://vision.middlebury.edu/MRF`.

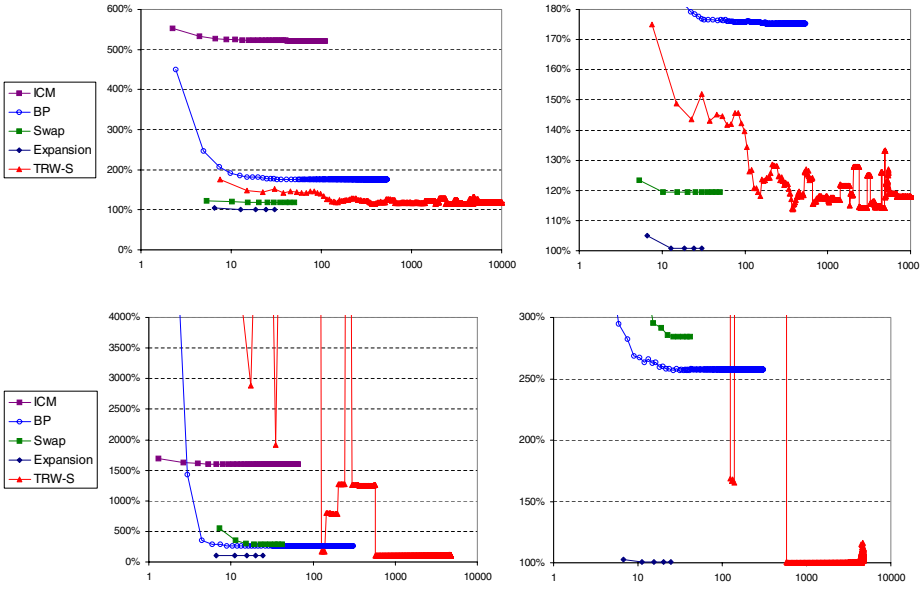


Fig. 2. Results on the Photomontage benchmarks, "Panorama" is at top and "Family" is below. Each plot shows energies vs. run time in seconds, using a log scale for time. The plots on the right are zoomed versions of the plots on the left. ICM is omitted in the right plots, due to poor performance. The associated color images can be found on the project web page.

in seconds. Note that some figures use a log scale for running time, which is necessary since some algorithms perform very poorly. For the y-axis, we made use of TRW's ability to compute a lower bound on the energy of the optimal solution. We normalized the energy by dividing it by the best known lower bound given by any run of TRW-S. Due to space limitations we had to omit the plots for the image restoration benchmark; they can be found on the project web page.

For all of these examples, the best methods achieved results that are extremely close to the global minimum, with less than 1 percent error. For example, on "Tsukuba", expansion moves and TRW-S got to within 0.27% of the optimum, while on "Panorama" expansion moves was within 0.78%. These statistics may actually understate the performance of the methods; since the global minimum is unknown, we use the TRW-S lower bound, which (of course) can underestimate the optimal energy.

The individual plots show some interesting features. In figure 1, TRW-S does extremely well, but in the "Teddy" energy it eventually oscillates. However, during the oscillation it achieves the best energy of any algorithm on any of our stereo benchmarks, within 0.018% of the global minimum. The same oscillation is seen in figure 2, though this time without as good performance. On the binary image segmentation problems, shown in figure 3, graph cuts are guaranteed to

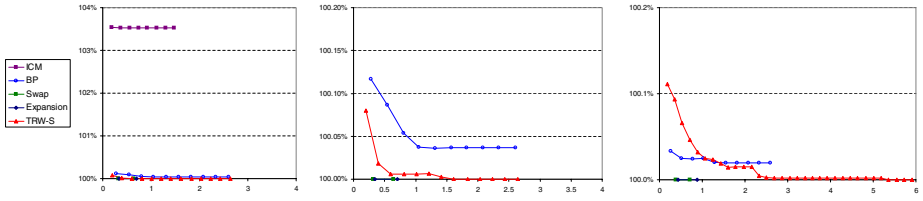


Fig. 3. Results on binary image segmentation benchmarks. Graph cuts are guaranteed to compute the global minimum, as is TRW-S. (In fact, this minimum is found by the first iteration of the swap move algorithm, which is equivalent to a single max flow computation.) Note that LBP comes extremely close (under 0.04% error), but never actually attains it.

Fig. 4. Results on "Panorama" benchmark. LBP output is shown at left, TRW-S in the middle, and expansion moves at right. Larger versions of these images are available on the project web page.

compute the global minimum, as is TRW-S (but not the original TRW [30]). LBP comes extremely close (under 0.04% error), but never actually attains it.

For reasons of space, we have omitted most of the actual images from this paper (they are available at `http://vision.middlebury.edu/MRF`). In terms of visual quality, the ICM results looked noticeably worse, but the others were difficult to distinguish on most of our benchmarks. The exception was the Photomontage benchmarks. On "Panorama", shown in figure 4, LBP makes some major errors, leaving slices of several people floating in the air. TRW-S does quite well, though the some of its seams are more noticeable than those produced by expansion moves (which gives the visually best results). On "Family" (not shown), LBP also makes major errors, while TRW-S and expansion moves both work well.

6 Discussion

The strongest impression that one gets from our data is of how much better modern energy minimization methods are than ICM, and how close they come to computing the global minimum. We do not believe that this is purely due to flaws in ICM, but simply reflects the fact that the methods used until the late 1990's performed poorly. (As additional evidence, [4] compared the energy produced by graph cuts with simulated annealing, and obtained a similarly large improvement.) We believe that our study demonstrates that the state of the art in energy minimization has advanced significantly in the last few years.

There is also a dramatic difference in performance among the different energy minimization methods on our benchmarks, and on some of the benchmarks there are clear winners. On the Photomontage benchmark, expansion moves perform best, which provides some justification for the fact that this algorithm is used by various image stitching applications [12, 26]. On the stereo benchmark, the two best methods seem to be TRW-S and expansion moves. There are also some obvious paired comparisons; for instance, there never seems to be any reason to use swap moves instead of expansion moves. In terms of runtime, the expansion move algorithm is clearly the winner among the competitive methods (i.e., all except ICM), but as noted not all methods have been optimized for speed equally.

There is clearly a need for more research on message-passing methods such as TRW-S and LBP. While LBP is a well-regarded and widely used method, on our benchmarks it performed surprisingly poorly (the only method it consistently outperformed was ICM). This may be due to a quirk in our benchmarks, or it may reflect issues with the way we scheduled message updates (despite the help we were given by several experts on LBP). TRW-S, which has not been widely used in vision, gave consistently strong results. In addition, the lower bound on the energy provided by TRW-S proved extremely useful in our study. For a user of energy minimization methods, this lower bound can serve as a confidence measure, providing assurance that the solution obtained has near-optimal energy. Another area that needs investigation is the use of graph cut algorithms for wider classes of energy functions than the limited ones they were originally designed for. The benchmarks that were most challenging for the expansion move algorithm (such as "Venus") use a V that is not a metric.

Another important issue centers around the use of energy to compare energy minimization algorithms. The goal in computer vision, of course, is not to compute the lowest energy solution to a problem, but rather the most accurate one. While computing the global minimum was shown to be NP-hard [4], it is sometimes possible for special cases. For example, the energy minimization problem can be recast as an integer program, which can be solved as a linear program; if the linear program's solutions happen to be integers, they are the global minimum. This is the basis for the approach was taken by [31], who demonstrated that they could compute the global minimum for several common energy functions on the Middlebury images. The global minimum has only slightly lower energy than that produced by graph cuts or LBP. In addition, [31] points out that the globally minimum is no more accurate than the results achieved with graph cuts or LBP. More precisely, according to [31] at best graph cuts produces an energy that is 0.018% over the global minimum, while at worst the energy is 3.6% larger; at best LBP gives an energy that is 3.4% higher, and at worst 30%.

In light of these results, it is clear that for the models we have considered, better minimization techniques are unlikely to produce significantly more accurate labelings. For the Middlebury stereo benchmarks this is particularly clear: the best methods produce energies that are extremely close to the global minimum; the global minimum, when known, is no more accurate than the ground truth; and, in fact, the ground truth has substantially higher energy. However, it

is still important to compare energy minimization algorithms using the energy they produce as a benchmark. Creating more accurate models will not lead to better results if good labelings under these models cannot be found. It is also difficult to gauge the power of a model without the ability to produce low energy labelings.

7 Conclusions and Future Work

There are many natural extensions to our work that we are currently pursuing, including energy minimization algorithms, classes of energy functions, and selection of benchmarks. While most of the energy minimization algorithms we have implemented are fairly mature, there is probably room for improvement in our implementation of LBP, especially in terms of the schedule of message updates. We also plan to implement several other modern algorithms, as well as additional benchmarks. We are particularly interested in [31], whose method could potentially achieve the global minimum on some of our benchmarks, and [32], who generalize the expansion move algorithm so that (like TRW) it also computes a lower bound on the energy.

We also plan to increase the class of energy functions we consider. We hope to investigate different grid topologies (such as the 8-connected topology for 2D, or 26-connected for 3D), as well as non-local topologies such as those used with multiple depth maps [15]. Finally, we will expand our set of benchmarks to include both more images and more applications, and continue to update our project web page to include the latest results in this rapidly evolving area.

References

1. Besag, J.: On the statistical analysis of dirty pictures (with discussion). Journal of the Royal Statistical Society, Series B **48** (1986) 259–302
2. Geman, S., Geman, D.: Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. IEEE Trans Pattern Anal Mach Intell **6** (1984) 721–741
3. Barnard, S.: Stochastic stereo matching over scale. Intern Journ Comp Vis **3** (1989) 17–32
4. Boykov, Y., Veksler, O., Zabih, R.: Fast approximate energy minimization via graph cuts. IEEE Trans Pattern Anal Mach Intell **23** (2001) 1222–1239
5. Kolmogorov, V., Zabih, R.: What energy functions can be minimized via graph cuts? IEEE Trans Pattern Anal Mach Intell **26** (2004) 147–59
6. Yedidia, J.S., Freeman, W.T., Weiss, Y.: Generalized belief propagation. In: NIPS. (2000) 689–695
7. Scharstein, D., Szeliski, R.: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. Intern Journ Comp Vis **47** (2002) 7–42
8. Geman, S., Graffigne, C.: Markov Random Field image models and their applications to computer vision. In: Intern Cong of Mathematicians. (1986) 1496–1517
9. Chou, P.B., Brown, C.M.: The theory and practice of Bayesian image labeling. Intern Journ Comp Vis **4** (1990) 185–210

10. Boykov, Y., Jolly, M.P.: Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In: ICCV. (2001) I: 105–112
11. Rother, C., Kolmogorov, V., Blake, A.: "GrabCut" - interactive foreground extraction using iterated graph cuts. SIGGRAPH **23** (2004) 309–314
12. Agarwala, A., et al.: Interactive digital photomontage. SIGGRAPH **23** (2004) 292–300
13. Tappen, M.F., Freeman, W.T.: Comparison of graph cuts with belief propagation for stereo, using identical MRF parameters. In: ICCV. (2003) 900–907
14. Boykov, Y., Kolmogorov, V.: An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. IEEE Trans Pattern Anal Mach Intell **26** (2004) 1124–1137
15. Kolmogorov, V., Rother, C.: Comparison of energy minimization algorithms for highly connected graphs. In: ECCV. (2006)
16. Li, S.: Markov Random Field Modeling in Computer Vision. Springer-Verlag (1995)
17. Efros, A.A., Freeman, W.T.: Image quilting for texture synthesis and transfer. SIGGRAPH (2001) 341–346
18. Kwatra, V., Schodl, A., Essa, I., Turk, G., Bobick, A.: Graphcut textures: Image and video synthesis using graph cuts. SIGGRAPH (2003)
19. Greig, D., Porteous, B., Seheult, A.: Exact maximum a posteriori estimation for binary images. Journal of the Royal Statistical Society, Series B **51** (1989) 271–279
20. Ishikawa, H.: Exact optimization for Markov Random Fields with convex priors. IEEE Trans Pattern Anal Mach Intell **25** (2003) 1333–1336
21. Hochbaum, D.S.: An efficient algorithm for image segmentation, Markov Random Fields and related problems. Journal of the ACM (JACM) **48** (2001) 686–701
22. Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient belief propagation for early vision. In: CVPR. (2004) 261–268
23. Kolmogorov, V.: Convergent tree-reweighted message passing for energy minimization. In: AISTATS. (2005)
24. Rother, C., Kumar, S., Kolmogorov, V., Blake, A.: Digital tapestry. In: CVPR. (2005)
25. Pearl, J.: Probabilistic reasoning in intelligent systems: networks of plausible inference. Morgan Kaufmann (1988)
26. Agarwala, A., et al.: Panoramic video textures. SIGGRAPH **24** (2005) 821–827
27. Snow, D., Viola, P., Zabih, R.: Exact voxel occupancy with graph cuts. In: CVPR. (2000) 345–352
28. Buehler, C., et al.: Minimal surfaces for stereo vision. In: ECCV. (2002) 885–899
29. Kolmogorov, V., Criminisi, A., Blake, A., Cross, G., Rother, C.: Bi-layer segmentation of binocular stereo video. In: CVPR. (2005) 407–414
30. Wainwright, M.J., Jaakkola, T.S., Willsky, A.S.: MAP estimation via agreement on (hyper)trees: Message-passing and linear-programming approaches. IEEE Trans Info Theory **51** (2005)
31. Meltzer, T., Yanover, C., Weiss, Y.: Globally optimal solutions for energy minimization in stereo vision using reweighted belief propagation. In: ICCV. (2005)
32. Komodakis, N., Tziritas, G. A new framework for approximate labeling via graph cuts In: ICCV. (2005)